On The Combination of Feature and Instance Selection

Jerffeson Teixeira de Souza, Rafael Augusto Ferreira do Carmo and Gustavo Augusto Campos de Lima
Universidade Estadual do Ceará
Brazil

1. Introduction

In the last decades, huge amounts of data became omnipresent in diverse areas of knowledge, such as business, astronomy, biology, and so on. Machine Learning and Knowledge Discovery in Databases (KDD) are fields in Computer Science that focus on the task of transforming these data into useful knowledge. In (Fayyad et al., 1996), KDD is defined as "*the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data*". Feature and Instance Selection belong to the practice of data preparation (or pre-processing), which is a preliminary process that transforms raw data into a format that is convenient to the data mining (or machine learning) algorithm.

Usually, data is stored in a *table-like* format: the columns of these tables are the *attributes* or *features* - they describe the data - and the rows, or lines, are the *records* or *instances* - they are the examples of the concept stored in the data. Feature and Instance selection processes allow applications, such as classification or clusterization, to focus only on the important (or relevant) attributes and records to the specific concept that is in study.

As important machine learning problems, Feature and Instance Selection have been studied systematically over the last decades, when several algorithms for solving them individually have been proposed. Such selection problems play a fundamental role in the pre-processing step of any learning task. By removing noise, irrelevant and redundant features and instances, and reducing the overall dimensionality of a dataset, feature and instance selection have been demonstrated to improve the performance of most machine learning algorithms, speed up the output of models and allow algorithms to deal with datasets whose sizes are gigantic. Even though the specialized literature have exhibited remarkable results in solving both the feature and instance selection problems individually, little work has been done to manage these solutions to work together in order to solve these related problems simultaneously or even understand the relationship between features and instances.

This chapter initially discusses the feature and instance selection problems and their relevance to machine learning, giving an accurate definition of both problems. Next, it surveys different approaches for dealing with feature selection and instance selection separately and some works that tried to integrate the solutions for these two problems, demonstrating the unexplored potential of such combination. Following the single and multi-objective models to these problems, it is presented and evaluated a metaheuristic-based framework for integrating the problems. Several experimental results demonstrate the interesting performance of the framework when compared to other standalone and combinational approaches over several natural datasets collected in the literature. Some conclusions and ideas for future works are given in the end of the chapter.

1.1 Problem´s Definition

In this chapter we are going to use the following formalization when referring to *datasets*, *features*, and *evaluation functions*. Based in (John et al., 1994) "each instance **X** is an element on the set $F_1 \times F_2 \times \ldots F_m$, where F_i is the domain of the *i*th feature". A dataset *D* is a set of tuples <**X**, C> where C is the class value of this example.
Given a classifier *C* and a dataset *D*, we define *G(C, D)* as a function that measures the *error rate* of this classifier on this dataset *D*.

1.2 The Feature Selection Problem

The Feature Selection problem involves discovering a subset of features such that a classifier built only with this subset would have better predictive accuracy than a classifier built from the entire set of features. Other benefits of feature selection include a reduction in the amount of training data needed to induce an accurate classifier, that is consequently simpler and easier to understand, and a reduced execution time. In practice, feature selection algorithms will discover and select features of the data that are relevant to the task to be learned.
In addition to irrelevant features, feature selection researchers have identified other examples of problematic features which may have a negative impact on the performance of learning systems such as redundant features and randomly class-correlated features. Irrelevant features are those that do not contribute to the predictive accuracy of a particular target concept. Redundant features refer to those that, even when relevant to a target concept, provide mostly information already present in another feature and, in fact, do not contribute to getting better predictors. Randomly class-correlated features are correlated to the target class most of the time, and random otherwise. Thus, irrelevant, redundant and randomly class-correlated features are worthless and removing them can improve the learning process. In fact, the feature selection process can be seen alternatively as the process of identifying and removing as many irrelevant, redundant and randomly class-correlated features as possible.
Then we can formulate the problem of feature selection as:

$$\begin{array}{c} Max\ G \\ Subject\ to \\ |F'| \geq 1 \end{array} \qquad (1)$$

A multiobjective version of it can seen as

$$\begin{array}{c} Max\ G\ ,\ Min\ \ |F| \\ Subject\ to \\ |F'| \geq 1 \end{array} \qquad (2)$$

Clearly a classifier built with a set of features $F' \subseteq F$ which is more accurate than one built with the whole set F is more interesting to use. Additionally the smaller it is the less computationally expensive it is. This characteristic is very important due to the datasets with high number of features found nowadays.

1.3 The Instance Selection Problem

The Instance Selection problem is basically the orthogonal version of the Feature Selection problem, as it involves discovering a subset of instances such that a classifier built only with this subset would have better predictive accuracy than a classifier built from the entire set of instances. In (Liu & Motoda, 2002), this problem is defined as "*to choose a subset of data to achieve the original purpose of a data mining application as if the whole data is used*". Clearly, instance selection cleans the dataset that is in use: it removes irrelevant examples, as well noisy and redundant ones. Instance Selection plays, consequently, two important roles: to improve computational efficiency, since the learning algorithm will consider only a subset of the original data, and to allow the induction of better classifiers (Blum & Langley, 1997).
Let's define a function $Freq(D^*, c)$ that calculates the frequency of the class c in the given dataset D*. A Δ is a value in the interval [0..1]. Given these two definitions, the initial dataset D, a generic subset of it D' and I the set of instances in this dataset then we can formulate the problem of instance selection as

$$\begin{array}{c} Max\ G \\ Subject\ to \\ \forall c \in C, Freq(D',c) - \Delta \leq Freq(D,c) \leq Freq(D',c) + \Delta \\ |I'| \geq 1 \end{array} \tag{3}$$

A multiobjective version of it can seen as

$$\begin{array}{c} Max\ G\ ,\ Min\ |I| \\ Subject\ to \\ \forall c \in C, Freq(D',c) - \Delta \leq Freq(D,c) \leq Freq(D',c) + \Delta \\ |I'| \geq 1 \end{array} \tag{4}$$

Like in the feature selection problem, a classifier built with a set of instances $I' \subseteq I$ which is more accurate than one built with the whole set I is more interesting to use. Additionally the smaller it is the less computationally expensive it is.

2. Related Works

Up to this date, several solutions have been proposed to deal with the feature and instance selection problems. In this section we briefly describe some important algorithms that work on each problem separately and show some approaches that handle both problems in a simultaneous way.

2.1 On Feature Selection

Well known feature selection algorithms perform very differently in identifying and removing irrelevant, redundant and randomly class-correlated features. Feature weighting algorithms such as Relief (Kira & Rendell, 1992), for instance, usually cannot identify redundant features since they evaluate features individually, not in sets of features like other feature selection algorithms. However, they are often very efficient in estimating feature relevance. Relief also suffers with randomly class-correlated features. In (Dash & Liu, 1997), the authors report that Relief preferred a correlated feature rather than a relevant one in the CorrAL dataset. The Focus algorithm (Almuallim & Dietterich, 1991), on the other hand, deals really well with irrelevant, redundant and class-correlated features since it looks for subsets that generate no inconsistency. Unless the redundant or class-correlated features perfectly duplicate their pairs (another feature or class label, respectively), they will be eventually eliminated by Focus. The LVF algorithm (Liu & Setiono, 1996) presents a similar behavior, since it will search for more consistent subsets. For small datasets or given enough time, LVF tends to get rid of undesirable features. Other results regarding the ability of feature selection algorithms in dealing with these problematic features can be found in (Dash & Liu, 1997). In this paper, the authors report results of several well known selection algorithms over three datasets (CorrAL, Parity3+3 and Monk3) containing together irrelevant, redundant and randomly class-correlated features.

As for the use of metaheuristics for the Feature Selection problem, authors have tried different approaches. A Genetic Algorithm-based feature selector (GA) proposed in (Vafaie & De Jong, 1992) applies a simple genetic algorithm to search through the subsets of features. Other examples of applications of genetic algorithms for feature selection can be found in (Beritelli et al., 2005) and (Sun et al., 2002). In (Tahir et al., 2004), the authors report the use of Tabu Search to select attributes to improve the classification of prostate needle biopsies. The paper reports a reduction of 50% in the classification error rate due to the proposed approach. The Simulated Annealing metaheuristic was used in (Filippone et al., 2006) to develop the SAIS (Simulated Annealing Input Selection) algorithm. Good experimental results were reported when using several datasets, including two bioinformatics datasets.

In (Souza, 2004) the author describes and discusses dozens of feature selection algorithms and expands a framework proposed earlier by (Dash & Liu, 1997) which classifies several algorithms according to their generation procedure and evaluation criterion. All these algorithms can be classified into three broad categories: *filters*, *wrappers* and *hybrid approaches*. *Filters* are those algorithms which perform the selection of features using an evaluation measure that classify the "quality" of these elements to differentiate classes without making use of any machine learning algorithm. *Wrappers* explicitly make use of machine learning algorithms in order to perform this measurement. Usually, *filters* are much less computationally expensive than *wrappers* but they produce subsets with less quality than those produced by *wrappers*. *Hybrid approaches* combine the best characteristics of both approaches, trying to produce very good subsets efficiently.

2.2 On Instance Selection

Most of the works on instance selection have been based on Nearest Neighbor classification.
In (Hart, 1968), the author proposed the Condensed Nearest Neighbor Rule (CNN), which finds a subset such that every member of the original dataset is closer to a member of the subset of the same class than to a member of the subset of a different class. This approach was extended in (Ritter et al., 1975) in the Selective Nearest Neighbor Rule (SNN) where every member of the original dataset must be closer to a member of the dataset of the same

class than to any member of the original dataset of a different class. The Reduced Nearest Neighbor Rule (RNN) was proposed in (Gates, 1972). It removes each instance if such a removal does not cause any other instances to be misclassified by the instances remaining.
In (Cano et al., 2003), the authors describe and evaluate four evolutionary approaches, including genetic algorithms, for the instance selection problem and report better data reduction percentages and higher classification accuracy in the experimental evaluation when using these approaches. Another application of genetic algorithms can be found in (Ramirez-Cruz et al., 2006). A description and comparison of several instance selection algorithms can be found in (Jankowski & Grochowski, 2004).
Like feature selection algorithms, instance selection algorithms can be classified in those three broad categories.

2.3 On the Combination of Feature and Instance Selection

The most natural and straight-forward way to combine feature and instance selection is to perform one process after the other. In practice, that has been the way the two problems have been integrated. Let's consider as FSIS, the application of a feature selection process followed by an instance selection process, and ISFS the opposite. Since these approaches are general, in the sense that they can be applied to any domain, we will use them as comparison base in our experiments.
Besides this approaches, in (Fragoudis et al., 2002) the authors propose the FIS (Feature and Instance Selection) algorithm, which targets both problems simultaneously in the context of text classification. It considers a set of documents, classified in one of two classes C and C', which contain a group of words each and operates in two steps. In the first step, it searches for a subset of the original vocabulary that contains the words that are the best predictors of the given class C. Next, only the documents which contain at least one word from this subset are kept. The second step searches, similarly, on the resulting dataset for a subset of words that are the best predictors of class C'. The output of the algorithm FIS contain the two subsets of features over the resulting documents from the first step. The authors reported a great decrease in the number of feature and training instances. It terms of accuracy, the algorithm, using the Naive Bayes classifier, performed in some cases equally or more accurate them SVM.
Some works also use metaheuristics to solve these two problems. In (Souza et al., 2008) the authors use two simultaneous Simulated Annealing (Kirkpatrick et al., 1983) runs to solve each problem separately but use the actual solution of each process to calculate the quality of both of them. There has been made a lot of work using genetic and evolutionary algorithms. It is quite natural to design the solution to these two problems as a chromosome which is the vector of all feature and all instances and then run a genetic algorithm to solve these problems. In (Ramirez-Cruz et al., 2006) a simple approach that splits the chromosome in two areas, the one of features which are coded as real values in [0..1] to weight the features, and the area of instances which are coded as boolean values to select the instances is presented. In (Kuncheva & Jain, 1999) the authors use boolean value coding to select feature and instances. The objective function used is the composition of the precision of 1-nn plus a value that penalizes the cardinality of each set. In (Sierra et al., 2001) the authors apply an adaptation of genetic algorithms, called Estimation of Distribution Algorithm (EDA), to select instances and features in the problem of estimating the likelihood of cirrhotic patients to die in at most 6 months after the interventional treatment called

Transjugular Intrahepatic Portosystemic Shunt (TIPS). In (Chen et al., 2005) it is made a study using an explicit multi-objective design to the problems of feature and instance selection, in which the goal is to maximize the performance of the 1-nn classifier and minimize both the number of attributes and instances. In (Ros et al., 2007) the authors model the problem in a multi-objective approach and solve them by a two-phase genetic algorithm. In (Ishibuchi & Nakashima, 2000) the authors use a genetic algorithm which is biased to decrease the number of features selected, by giving a bigger probability to the changing that exclude features from the solutions.

3. A Framework for Simultaneous and Independent Feature and Instance Selection

A deeper look into the related works on the combination of feature and instance selection shows some points already addressed by these solutions and new questions. The first point is that the majority of the approaches which try to solve these problems in a broad generic formulation solve them by using specialized versions of genetic algorithms, trying to separate the chromosome into two different areas, one for features and one for instances, and applying separate operators to each area. These approaches arise from the natural easiness of modeling the solutions to these problems as chromosomes and the need to cope with these two different problems separately. The other point addressed is that either these approaches work on a specific field of supervised learning (e.g. text mining) or depend on a specific classifier (e.g. kNN). The reader may question "How can we use other metaheuristics beyond genetic algorithms to solve these two problems?" or "How could we build a general framework to with them simultaneously?". This section tries to give answers to these questions.
Here we describe an extension of the work presented in (Souza et al., 2008) as a general metaheuristics-based framework for simultaneous and independent feature and instance selection. This framework is an effort to build an algorithm that can deal with both problems simultaneously, since these problems are clearly related to one another and the work made to select a subset of features can also be reused to select instances (and vice-versa) but they are also independent, meaning that the algorithms that solve one of these problems do not have to tackle the other one as well. The key idea here is to provide a *joint subset evaluation*, in which the quality of a subset of features depends on the quality of a subset of instances (and vice-versa). This means that although the search processes are independent, they are guided by this *joint evaluation function*, which gives what we call a "power of influence" of each solution of the separate problems over the other.

3.1 The Framework for Feature and Instance Selection

In order to make definitions clear, we must explain that this framework works with two different solutions for each problem: the *best* and the *actual* solution. The *best* solution is, as the name itself explains, the solution which achieved the best evaluation value so far in the search process. The *actual* solution is the solution generated at every iteration to be tested to see whether it is better than the *best* solution so far. In some metaheuristics, like Simulated Annealing, the process of search does not depend on the *best* solution, although this solution is stored, but in others like VNS (Mladenović & Hansen, 1997) the *best* solution is the one which guides the search.

When applied to feature and instance selection, search metaheuristics can be seen as wrappers, as they generate subsets (solutions), evaluate them using some classifier to test whether they are good solutions or not and guide the search process by this evaluation value achieved by each solution. This version of the framework works basically controlling this evaluation process of each subset generated by the search. The framework can be described as follows in figure 1

Framework for Mono-Objective Simultaneous and Independent Feature and Instance Selection
Input: *Dataset* dt, *Feature Selection Algorithm* fs, *Instance Selection Algorithm* is, *Evaluation Function* ef
Output: *Dataset* ndt

```
1.  While(Has Iterations(fs) || Has Iterations(is))
2.  Do
3.  $fs_{ss}$ = Next Solution(fs, dt)
4.  $is_{ss}$ = Next Solution(is, dt)
5.  eval = Evaluation(ef, dt, $fs_{ss}$, $is_{ss}$)
6.  Update(fs, eval, $fs_{ss}$)
7.  Update(is, eval, $is_{ss}$)
8.  Done
9.  ndt = Create Subset(dt, Best Subset(fs), Best Subset(is))
10. Return ndt
```

Fig. 1. The Framework

In this framework, the relationship between these entities, features and instances, is treated as something related to their quality to a supervised learning task. This means that the quality of features used in the supervised learning task is intrinsically related to the examples that represent the concept to be learned, and vice-versa. Examples are only considered "good" ones if they are described by attributes that represent the concept to be learned clearly, and features are only important if they capture this concept present in these examples. This is the justification to the presented approach.
A textual description of the framework can be seen as: Initially the complete sets of features and instances are set as initial solutions. There are two separated processes for selecting features and instances. The main loop started in the line number 1 controls the search processes. Starting in line number 3 (4) the new solution is generated using the metaheuristic for feature (instance) selection. New solutions are generated only when the *Has Iterations* test has *true* value, otherwise the *Next Solution* function must return the best solution found in the search process. In line 5 the new solutions are evaluated. This step is the *joint evaluation function* that works by getting the *actual solutions* from the feature selection and instance selection processes, then creating a subset from the initial dataset by using these two subsets and then evaluate then using *k-fold* cross validation, for example. Finally the search processes are updated. This update is basically the exchange of solutions if the new one achieved a better evaluation than the old one and any other process needed by the metaheuristic like for example in Simulated Annealing, when even if the actual solution is worse than the best one, it can be the next which guides the following steps of the search. In line 9 the whole procedure is ended, and the subset generated by the two solutions is returned as a new dataset to the supervised learning task. The figure 2 shows a

graphical representation of the framework. The blue boxes are the *best solutions* in a given iteration of the processes. The red boxes are the *actual solutions* in them and as it can be seen, they are evaluated together using the function *G*. In some iteration they replace the *best solution* but in other ones they do not have better evaluation so the best solutions remain the same.

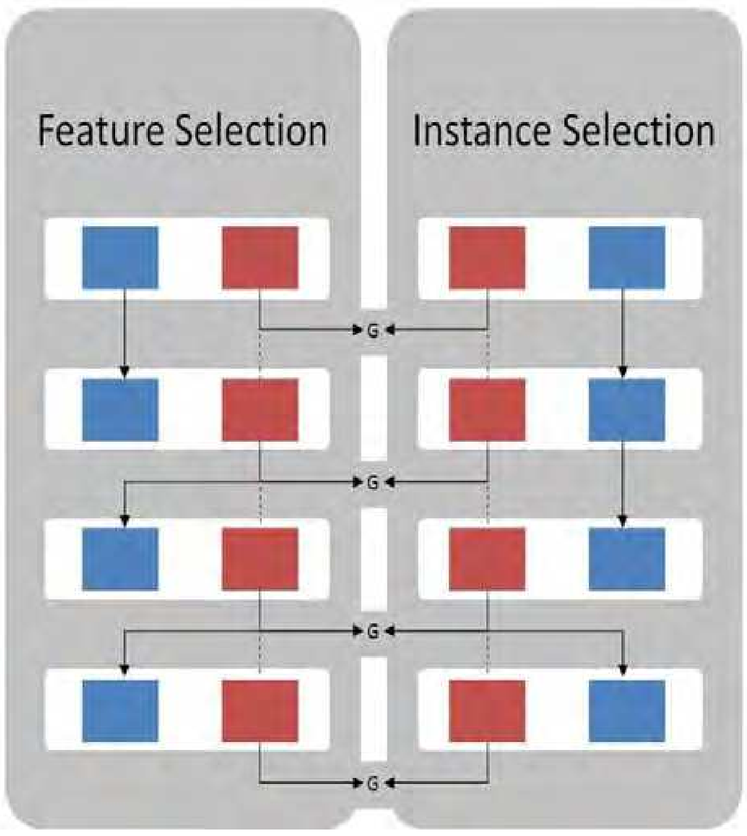

Fig. 2. A graphical view of the framework (Blue box – *best solution*; Red box – *actual solution*)

3.2 Extension to the Framework

The framework described in the last section is a basic view of it. An interesting extension can be made for handling populational metaheuristics.

Populational metaheuristics create, at each iteration, a set of new *actual* solutions. Then the evaluation of each new solution is calculated and operators of intensification and diversification are applied. If we remember the fact that in this framework the evaluation of a solution does not depend on itself solely, this fact adds the question of "Which solution from the other process should I use in the *joint evaluation function*?" or "How can I calculate the *best actual* solution in this given set of solutions?".

Our answers to these questions are quite simple. The answer to the first question is "the *best actual* solution from the last iteration". In the first iteration the whole set of features or instances is used to evaluate the new solutions and the search continues always reusing the best actual solution of the last iteration. By doing this, the searches are still guided by both solutions and only good solutions will guide this process. Nevertheless, the operators of *intensification* and *diversification* will work normally, without any loss to the search process.

The answer to the second question is "by using the *best actual* solution from the last iteration" as showed in the previous explanation.
Figure 3 gives a graphical explanation to the idea presented here. The reader must pay attention to the green arrows. They show that the *best actual* solution (the yellow one) is being used to evaluate the subsets of features (or instances) of the next generation. Besides, there are separate evaluation procedures to the searches. These are the biggest differences to the initial framework. Although there are these two separated procedures, the evaluations of the actual solutions still depend on the other search process. Once more as in the initial framework, in some iterations one of the solutions present in the actual population might replace the *best solution* found so far but in other ones they do not have better evaluation so the best solutions remains the same.

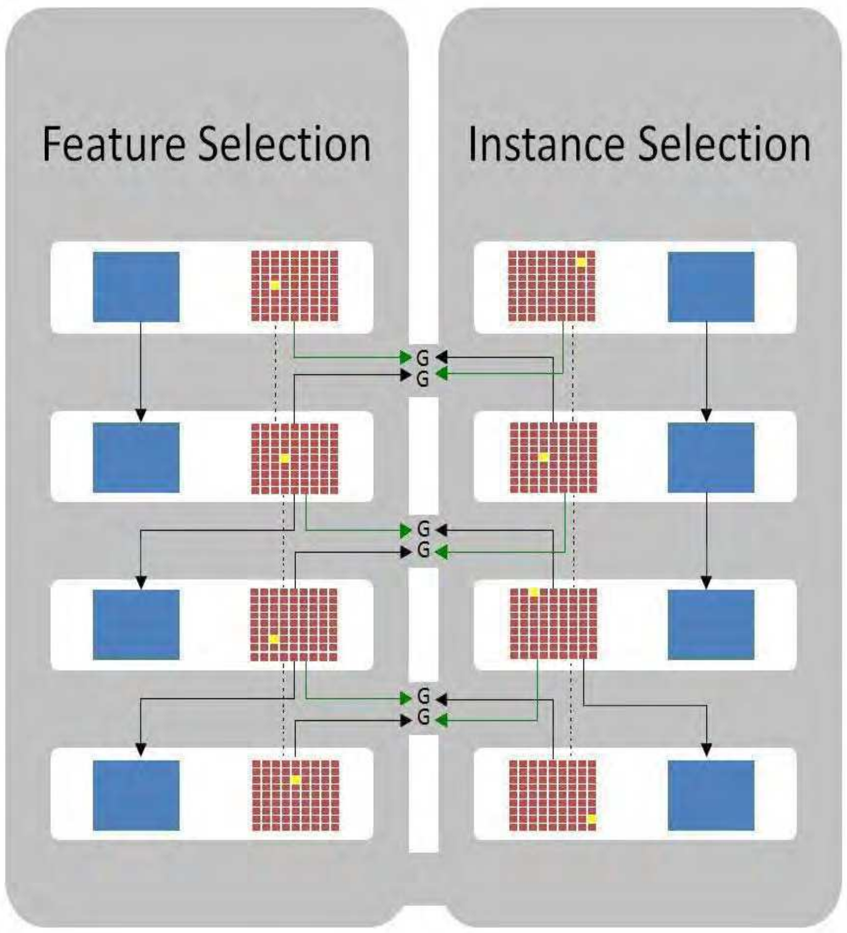

Fig. 3. A graphical view of the framework (Blue box – *best solution*; Red box – *actual solution;* Yellow box – *best actual solution*. Green arrow – The *best actual solution* is being used to evaluate the next generation of solutions)

4. Framework Evaluation

In this section we present and discuss the results obtained in several simulations executed in order to test the effectiveness of the proposed framework. This section tries to make clear the answer to the question "Is it worth using this framework?".

To make these simulations we have chosen some well-known datasets used for machine learning tasks found at the UCI Machine Learning Repository (Asuncion and Newman, 2007). These datasets are the Audiology (70 attributes, 226 instances), Autos (26, 205), Colic (23, 368), Credit (16, 690), Ionosphere (35, 351), Labor (17, 57), Lymph (19, 148), Primary-Tumor (18, 339), Sonar (61, 208), Soybean (36, 683) and Vote (17, 435).
We implemented seven different strategies to tackle with the feature and instance selection problems. The first one, here called **ind**, consists in making two separate selection processes and then joining the subsets generated by these processes in the end. The solution generated by the feature selection process and the other generated by the instance selection one are joined to create the subset only when the search processes are ended. The **fsis** is a sequential approach in which it is run a feature selection process followed by an instance selection process. The dataset used in the feature selection is the whole initial dataset, but in the dataset used by the instance selection process, only the best set of features found is used to represent the examples. The **isfs** approach follows the same idea, but now the first process is an instance selection and the second is a feature selection. Finally **comb** is the name given to the approach presented in the framework.
Some pieces of different software were used to make these simulations. From the Weka Data Mining Software (Witten & Frank, 2005) we used several classes to represent datasets, attributes and examples and to create and evaluate models. From jMetal Metaheuristics Framework (Durillo et al., 2006) we used some classes to represent the solutions to these problems and also some classes of metaheuristics. The Evaluation method chosen to be used in these simulations was a 10-fold cross-validation. The classifiers used were the C4.5, Naive Bayes and kNN.

4.1 Simulation Using the Simulated Annealing Metaheuristic

The results presented in this section are the same presented in the former work of (Souza et al., 2008). The architecture implemented in that work is the same of this general framework but it was implemented using the Simulated Annealing metaheuristic.
For this simulation we implemented the Simulated Annealing metaheuristic to use it in both selection problems. Simulated Annealing is a metaheuristic that consists in a randomized local search, which simulates the process of physical annealing. This physical process consists in heating a material to a desired temperature, followed by a slow cooling process. The first step gives energy to the atoms and they move randomly through states of high energy, changing the material's structure fast. The second step, which is performed slowly, gives them the chance to arrange themselves into a configuration of lower energy.
In analogy with the physical process, Simulated Annealing changes the actual solution to a neighbor solution, depending on the quality of this neighbor solution or the value of a function that is calculated in accordance with the temperature parameter, which decreases during the process.
The coding of solutions to this problem is basically an array of boolean values which has length equal to the number of features or instances. Using this coding, we defined that two solutions are considered *neighbors* only and if only they have at most 10% of bits set to different values, i.e., when applied a XOR operator to these to problems, the result contains only 10% of bits set to *true*.

	< 0.001	< 0.005	< 0.01
Comb vs IND	8 x 0	1 x 0	1 x 0
Comb vs FSIS	0 x 0	0 x 0	0 x 0
Comb vs ISFS	0 x 0	0 x 0	2 x 0

Table 1. Pairwise comparison between **comb** and other approaches

We have run all seven approaches described earlier in two different scenarios. In the first one, each approach was given an unlimited time to run and generate a solution. After two executions of each approach in every dataset, there were twenty *Error Rate* values available.

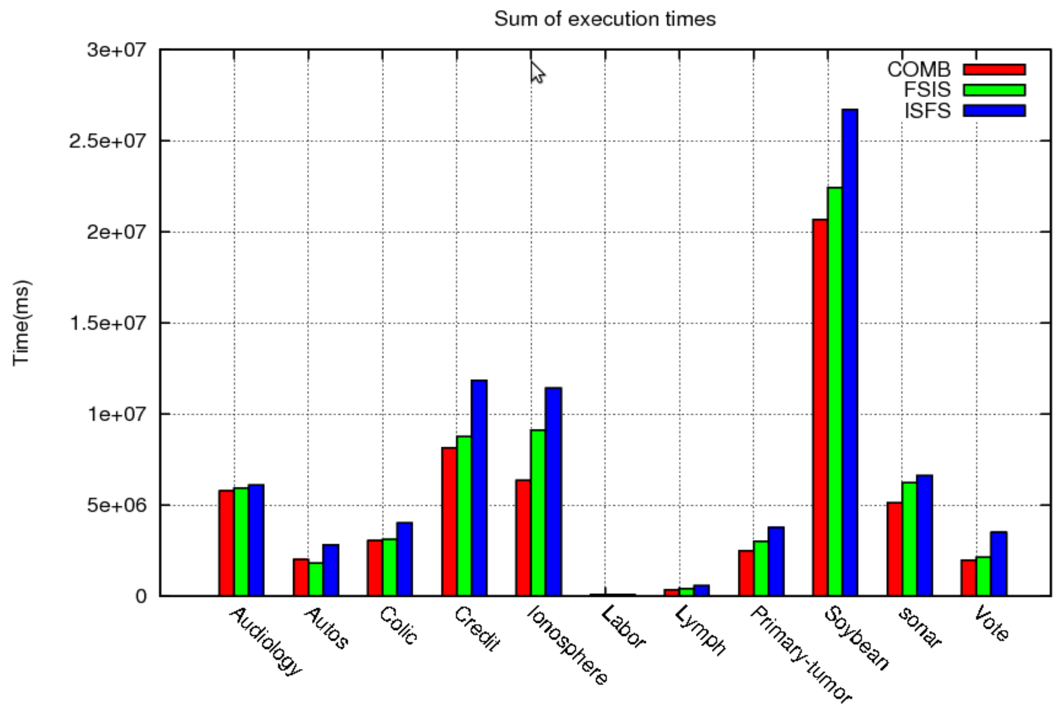

Fig. 4. The sum of execution times – Unlimited Time Scenario

Table 1 summarizes the results of a pairwise comparison between **comb** and the other approaches that solve both problems. The results represent the number of times each strategy outperformed the other, in terms of the accuracy of the final classifier, using the student's t-test with the corresponding confidence levels (0.001, 0.005 and 0.01).
Clearly the performance of **comb** is much better than the **ind** and slightly better than the other two approaches. So when talking about performance it is not clear why to use this approach. But looking at figure 3 we see that the **comb** approach usually requires less time to reach the best error rate in the datasets. This figure shows the sum of time of all tests executed by each approach.
In the second scenario we defined a limit to the execution time of every run. Figure 4 shows that when this constraint is added to the problem and this time is not enough to complete the search, the **comb** approach converges to low values of error rate faster than the other two approaches.

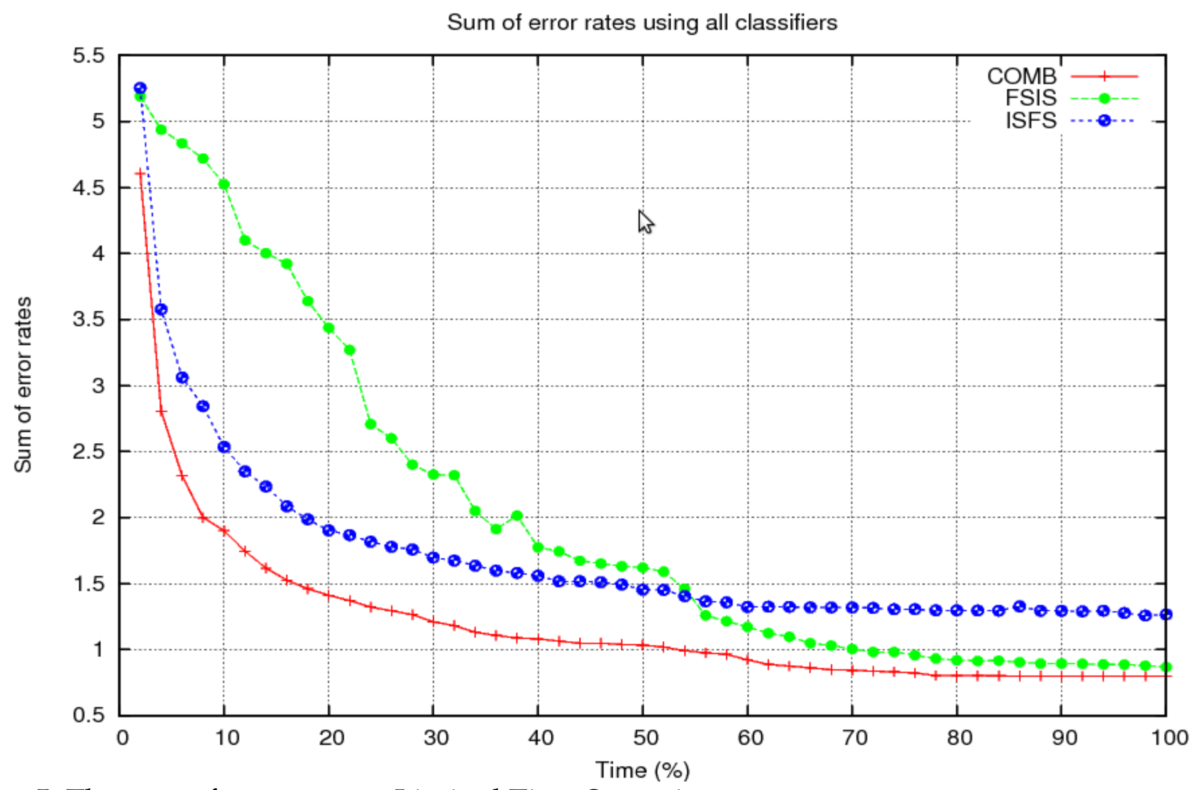

Fig. 5. The sum of error rates – Limited Time Scenario

5. Conclusion

In this chapter we discussed two important problems in the pre-processing step of many supervised learning tasks. A list of well-known algorithms were presented and discussed. A new framework was proposed, extending the concept proposed by the authors in a previous work. This framework was validated by some simulations using the metaheuristic Simulated Annealing and NSGA-II. These simulations show that although the quality of solutions generated by this framework is quite similar to those obtained by sequential executions, this approach reaches the better solutions faster than the other approaches.
The frameworks is based on what we called *"power of influence"*, i.e. the quality of features in a given supervised learning task is intrinsically related to the quality of instances used in this task, and vice-versa. Based on this we created the framework that work with two separated *wrappers* for these two problems, jointing them in a single evaluation procedure.

5.1 Future Work - The Framework for Multi-Objective Feature and Instance Selection

An important characteristic we want to add to this framework in the future is the possibility to handle the multi-objective versions of the two selection problems. The usage of multi objectives brings new power but also new problems to the search processes. In these formulations, the characteristic of *total ordering* is replaced by *partial ordering*, using the concept of *Pareto optimality*. The ideas of *better* and *worse* are replaced by *dominance*, *non-dominance*. Given two solutions a, b and a set of functions F to be minimized (or maximized, but in this explanation we suppose they are to be minimized), we say that a *weakly* dominates b if and only if

$$\forall f_i \in F, f_i(a) \leq f_i(b) \text{ and } \exists f_i \in F, f_i(a) < f_i(b) \quad (5)$$

The concept of *strong* dominance requires that

$$\forall f_i \in F, f_i(a) < f_i(b). \quad (6)$$

When there is a set of solutions in which none of them dominate or are dominated by the others, we say these solutions are in the *Pareto front*, i.e., they are solutions equally good, in a way that one cannot say *à priori* which one of them is the best one without making any other assumption.

This usage adds the same questions generated by populational metaheuristics, such as "Which solution from the other process should I use in the *joint evaluation function*?" or "How can I calculate the *best actual* solution in this given set of solutions if there will be some 'equally good' ones?".

The answers related to the multi-objective approaches seem to be similar to the ones given to populational metaheuristics but they weren't tested yet. Given that it is needed to evaluate all the subsets of features (and vice-versa), the algorithm can use any of the subsets of instances from the last iteration which are in the *Pareto front* since all of them are equally good. A reasonable solution would be to pick a random solution from the *Pareto front* of the instance selection process every time the algorithm has to evaluate a subset of features. This approach increases *diversification* because several different solutions are used to guide the search and there is no loss in *intensification* as only good solutions are used in this process.

In the end of the search processes there will be two *Pareto fronts*: one of features and one of instances. At this moment the user have several alternatives like choosing one solution of each search or generating all combinations of solutions and picking the one which is the best. How to deal with these two *Pareto fronts* is an open question so far.

6. References

Almuallim, H. & Dietterich, T. (1991). Learning with many irrelevant features. *Proceedings of the Ninth National Conference on Artificial Intelligence* (AAAI'91), pp. 547-552, AAAI Press, Anaheim

Asuncion, A. & Newman, D. J. (2007). *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mlearn/MLRepository.html], University of California, School of Information and Computer Science, Irvine, CA

Beritelli, F.; Casale, S.; Russo, A. & Serrano, S. (2005). A genetic algorithm feature selection approach to robust classification between "positive" and "negative" emotional states in speakers. *Thirty-Ninth Asilomar Conference on Signals, Systems and Computers*, pp. 550-553

Blum, A. & Langley, P. (1997). Selection of Relevant Features and Examples in Machine Learning. *Artificial Intelligence*, 97, 1-2, December 1997, 245-271, 0004-3702

Cano, J.; Herrera, F. & Lozano, M. (2003). Using evolutionary algorithms as instance selection for data reduction in kdd: an experimental study. *IEEE Transactions on Evolutionary Computation*, 7, 6, December 2003, 561-575, 1089-778X

Chen, J-H.; Chen, H-M. & Ho S-Y. (2005). Design of nearest neighbor classifiers: multi-objective approach, *International Journal of Approximate Reasoning*, 40, 1, July 2005, 3-22, 0888-613X

Dash, M. & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis – An International Journal*, 1, 131-156

Durillo, J. J.; Nebro, A. J.; Luna, F.; Dorronsoro, B. & Alba, E. (2006). *TechRep jMetal: a Java Framework for Developing Multi-Objective Optimization Metaheuristics*., Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, 2006

Fayyad, U.; Piatetsky-Shapiro, G. & Smyth, P. (1996). From data mining to knowledge discovery in databases. *Ai Magazine*, 17, 3, 37-54, 0738-4602

Filippone, M.; Masulli, F.; Rovetta, S. & Constantinescu, S. P. (2006). Input selection with mixed data sets: A simulated annealing wrapper approach. *Conferenza Italiana Sistemi Intelligenti* (CISI 06), Ancona

Fragoudis, D.; Meretakis, D. & Likothanassis, S. (2002). Integrating feature and instance selection for text classification. *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 501-506, 1-58113-567-X, ACM, New York

Gates, G. (1972). The reduced nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 18, 3, May 1972, 431-433, 0018-9448

Hart, P. (1968). The condensed nearest neighbor rule, *IEEE Transactions on Information Theory*, 14, 3, May 1968, 515-516, 0018-9448

Ishibuchi, H. & Nakashima, T. (2000). Multi-objective pattern and feature selection by a genetic algorithm, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1069-1076, Morgan Kaufmann

Jankowski, N. & Grochowski, M. (2004). Comparison of instances selection algorithms I. Algorithms Survey, In: *Artificial Intelligence and Soft Computing - ICAISC 2004*, Springer, 978-3-540-22123-4

John, G.; Kohavi, R. & Pfleger, K. (1994). Irrelevant features and the subset selection problem. *Proceedings of the Eleventh International Conference on Machine Learning (ICML'94)*, 121–129

Kira, K. & Rendell, L. (1992). The feature selection problem: Traditional methods and new algorithm, *Proceedings of the Tenth National Conference on Artificial lntelligence*, pp. 129-134, MIT Press

Kirkpatrick, S.; Gelatt, C. D. & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220, 4598, May 1983, 671-680, 00368075

Kuncheva, L. I. & Jain, L. C. (1999). Nearest neighbor classifier: simultaneous editing and feature selection, *Pattern Recognition Letters*, 20, 11, November 1999, 1149-1156, 0167-8655

Liu, H. & & Motoda, H. (2002). On issues of instance selection. *Data Mining and Knowledge Discovery*, 6, 2, 115-130, 1384-5810

Liu, H. & Setiono, R. (1996). A probabilistic approach to feature selection – A filter solution. *Proceedings of the Thirteenth International Conference on Machine Learning* (ICML'96), pp. 319-327, MIT Press

Mladenović, N. & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, 24, 11, November 1997, 1097-1100, 0305-0548

Ramirez-Cruz, J. F. ; Fuentes, O.; Alarcon-Aquino, V. & Garcia-Banuelos, L. (2006). Instance selection and feature weighting using evolutionary algorithms. *15th International Conference on Computing* (CIC '06), 73-79, 0-7695-2708-6

Ritter, G. ; Woodruff, H.; Lowry, S. & Isenhour, T. (1975). An algorithm for a selective nearest neighbor decision rule (corresp.). *IEEE Transactions on Information Theory*, , 21, 6, November 1975, 665-669, 0018-9448

Ros, F. ; Guillaume, S. ; Pintore, M. & Chrétien, J. R. (2007). Hybrid genetic algorithm for dual selection, *Pattern Analysis & Applications*, 11, 2, June 2008, 179-198, 1433-755X

Sierra, B. ; Lazkano, E. ; Inza, I. ; Merino, M. ; Larrañaga, P. & Quiroga, J. (2001), *Lecture Notes in Computer Science*, 2101/2001, 20-29, 978-3-540-42294-5

Souza, J. T. (2004). Feature selection with a general hybrid algorithm. *Doctoral dissertation, University of Ottawa, School of Information Technology and Engineering (SITE)*, Ottawa

Souza, J. T. ; Carmo, R. A. F. & Campos, G. A. L. (2008). A novel approach for integrating feature and instance selection. Proceedings of the International Conference on Machine Learning and Cybernetics, pp. 374-379, 978-1-4244-2095-7, July 2008

Sun, Z.; Bebis, G.; Yuan, X. & Louis, S. J. (2002). Genetic feature subset selection for gender classification: A comparison study. *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision* (WACV'02), IEEE Computer Society, Washington

Tahir, M.; Bouridane, A.; Kurugollu, F. & Amira, A. (2004). Feature selection using tabu search for improving the classification rate prostate needle biopsies. *Pattern recognition*, 2, 335-338, 0031-3203

Vafaie, H. & De Jong, K. (1992). Genetic algorithms as a tool for feature selection in machine learning. *Proceedings of the Fourth International Conference on Tools with Artificial Intelligence*, pp. 200-204, Arlington

Witten, I. H. & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*, Morgan Kaufmann, San Francisco, 2005

Machine Learning
Edited by Yagang Zhang

ISBN 978-953-307-033-9
Hard cover, 438 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Machine learning techniques have the potential of alleviating the complexity of knowledge acquisition. This book presents today's state and development tendencies of machine learning. It is a multi-author book. Taking into account the large amount of knowledge about machine learning and practice presented in the book, it is divided into three major parts: Introduction, Machine Learning Theory and Applications. Part I focuses on the introduction to machine learning. The author also attempts to promote a new design of thinking machines and development philosophy. Considering the growing complexity and serious difficulties of information processing in machine learning, in Part II of the book, the theoretical foundations of machine learning are considered, and they mainly include self-organizing maps (SOMs), clustering, artificial neural networks, nonlinear control, fuzzy system and knowledge-based system (KBS). Part III contains selected applications of various machine learning approaches, from flight delays, network intrusion, immune system, ship design to CT and RNA target prediction. The book will be of interest to industrial engineers and scientists as well as academics who wish to pursue machine learning. The book is intended for both graduate and postgraduate students in fields such as computer science, cybernetics, system sciences, engineering, statistics, and social sciences, and as a reference for software professionals and practitioners.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Jerffeson Teixeira de Souza, Rafael Augusto Ferreira do Carmo and Gustavo Augusto Campos de Lima (2010). On The Combination of Feature and Instance Selection, Machine Learning, Yagang Zhang (Ed.), ISBN: 978-953-307-033-9, InTech, Available from: http://www.intechopen.com/books/machine-learning/on-the-combination-of-feature-and-instance-selection